Folding Clothes Autonomously: A Complete Pipeline

Andreas Doumanoglou, Jan Stria, Georgia Peleka, Ioannis Mariolis, Vladimír Petrík, Andreas Kargakos, Libor Wagner, Václav Hlaváč, Tae-Kyun Kim, and Sotiris Malassiotis

***Abstract*—This work presents a complete pipeline for folding a pile of clothes using a dual-arm robot. This is a challenging task both from the viewpoint of machine vision and robotic manipulation. The presented pipeline comprises of the following parts: isolating and picking up a single garment from a pile of crumpled garments, recognizing its category, unfolding the garment using a series of manipulations performed in the air, placing the garment roughly flat on a work table, spreading it, and finally folding it in several steps. The pile is segmented into separate garments using color and texture information, and the ideal grasping point is selected based on the features computed from a depth map. The recognition and unfolding of the hanging garment are performed in an active manner, utilizing the framework of Active Random Forests to detect grasp points, while optimizing the robot actions. The spreading procedure is based on the detection of deformations of the garment contour. The perception for folding employs fitting of polygonal models to the contour of the observed garment, both spread and already partially folded. We have conducted several experiments on the full pipeline producing very promising results. To our knowledge, this is the first work addressing the complete unfolding and folding pipeline on a variety of garments, including T-shirts, towels, and shorts.**

***Index Terms*—clothes, perception, manipulation, deformable objects, active vision, random forests, robotics**

I. Introduction

Machine perception and automated manipulation of soft deformable objects has gained an increasing interest in cognitive robotics. The main challenge is in the large variation of their shape, which makes both their perception and manipulation very challenging. The problem is of high importance since many objects, which we interact with in our everyday life, are deformable. One important area of robotics research, where handling of soft deformable objects is ubiquitous, is household robots. The possible tasks comprehend laundering, wiping the dishes or hanging the curtains. The household robots can have a significant impact on our everyday life, particularly in aging societies of the developed countries [1].

The task of robotic laundering comprises several interesting problems in perceiving and manipulating clothes. The robot should be able to sort the dirty clothes according to their color and material, put them into the washing machine, move them into the drying machine, and finally fold the washed clothes. The presented paper deals with the latter challenge. More precisely, we propose the method for isolating a single item from the heap of crumpled washed garments, recognizing its type, unfolding the isolated garment, spreading it if necessary, and finally folding it while taking its type into account. The task is approached using a dual-arm robot with attached cameras and range sensors. The main challenge is that clothes are soft and highly deformable objects. This makes their perception very difficult since the same piece of garment can be posed in infinitely many configurations. The robotic manipulation of clothes is a challenging task as well because the garment is being further deformed while manipulated.

This work is based on our previous papers [2]–[5]. Two of them deal with the first part of the pipeline, which is active recognition and unfolding of the unknown garments [2], [3]. The other two focus on folding the garment which was already unfolded and spread on a table [4], [5]. Both methods are integrated into a single working pipeline in the proposed work. The pipeline is completed by adding two steps, which have not been described before. The first one is a novel method for grasping a single garment from a table before unfolding it. The second one is an intermediate spreading stage used in a case the unfolded garment is not adequately spread out on a table. Furthermore, we extensively evaluate the proposed pipeline by performing experiments with various garments. Fig. 1 shows a successful run of the pipeline applied to a T-shirt. As far as we know, this is the first working solution for bringing a heap of crumpled garments to the folded state. The closest to our work is [6], where authors demonstrated the complete folding task only for towels, whereas our approach can handle a variety of garments. Moreover, our approach is more computationally efficient, resulting in faster performance.

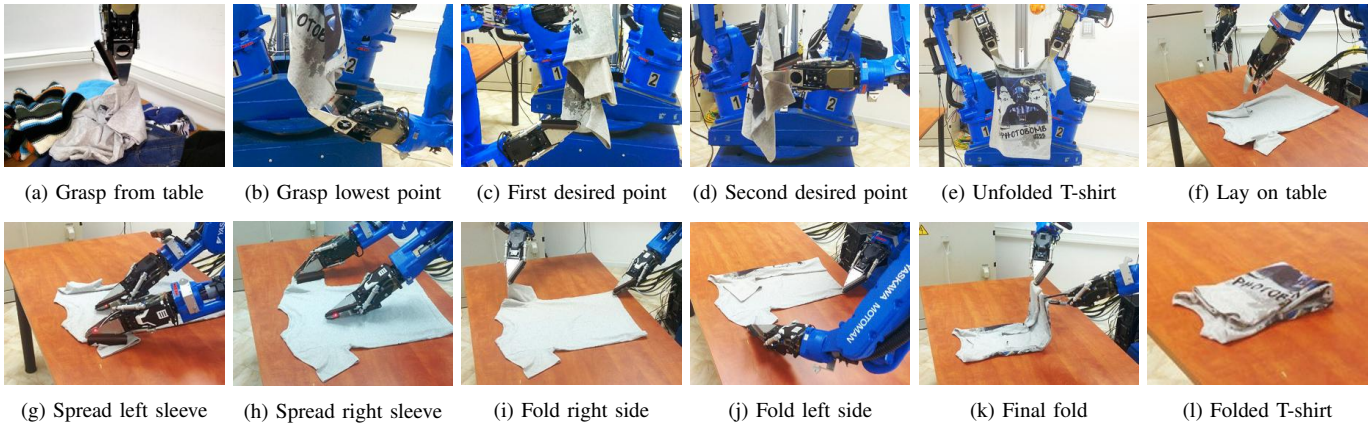
The main contributions of our work are in summary:

- A robust vision-based method for garment grasping.
- Clothes classification and unfolding performed in an active perception manner, based on our novel framework of *Active Random Forests* [3].
- A new method for spreading incompletely spread garments laid on the table.
- Model matching technique for clothes folding, based on dynamic programming [5].
- Implementation of the above methods into a complete folding pipeline, its deployment on a dual-arm robot and experimental evaluation using various garments.

The rest of the paper is organized as follows. Section II discusses the related work on machine perception and robotic manipulation of clothes, along with some active vision works. Section III describes how a single garment is grasped from the pile of crumpled garments and lifted up. Section IV deals

A. Doumanoglou and J. Stria contributed equally to the proposed work. J. Stria is the corresponding author: striajan@fel.cvut.cz.

 10.1109/TRO.2016.2602376



(a) Grasp from table (b) Grasp lowest point (c) First desired point (d) Second desired point (e) Unfolded T-shirt (f) Lay on table

(g) Spread left sleeve (h) Spread right sleeve (i) Fold right side (j) Fold left side (k) Final fold (l) Folded T-shirt

Fig. 1: The complete folding pipeline of a T-shirt. a) The optimal grasping point is selected, and the T-shirt is lifted from the table. b) Its lowest point is regrasped to reduce the number of possible configurations. c) The first desired point is grasped with the free arm, the lowest point is released, and d) the second desired point is grasped. e) The T-shirt is unfolded by stretching the grasped points. f) The unfolded T-shirt is laid on the empty table. g) Both left and h) right sleeve are spread using the brush tool if necessary. i) The folding procedure comprehends folding the right and j) left side of the T-shirt before k) performing the final fold. l) The result is the fully folded T-shirt.

with recognizing category of the hanging garment, estimating its pose and unfolding it step by step in a series of manipulations. Section V describes spreading of the not fully unfolded garment. Section VI presents methods for estimating the pose of the spread garment and folding it, while checking its state after each fold. Section VII demonstrates experimental evaluation of the complete pipeline and its individual parts. In this section, we also describe our dual-arm robot, which is used for the actual manipulation, both from the hardware and software view. We conclude our work and give ideas for possible future extensions in Section VIII.

II. Related work

The autonomous garment folding pipeline can be decomposed into several sub-tasks such as grasping a garment from a pile, its classification and pose estimation, unfolding, flattening, and folding. The majority of the existing works focus on one particular sub-task only. Therefore, we summarize the state of the art for each sub-problem separately in the following sub-sections. We also compare the existing methods to our approach.

A. Grasping

Robotic manipulation of clothes was pioneered by Hamajima and Kakikura [7], [8]. They propose the concept of the complete pipeline for bringing a heap of washed clothes to a stack of folded clothes. They focus mainly on isolating a single garment from the heap. The grasping point is selected from a certain uniform color region obtained after recursively segmenting the image of the heap. Grasping a single garment laid on a table was also approached by Ramisa *et al.* [9], [10]. The strategy is to select a highly wrinkled grasping point. The measure of wrinkledness is based on the classification of features computed from color and depth images [9]. The special descriptor called FINDDD, which is based on normals computed from depth data, is used in the extended work [10].

Cusumano-Towner *et al.* [11] first find the outer boundary of a crumpled garment by segmentation. The garment is then grasped from a side. Foresti *et al.* [12] address detection of grasping points on furs. After the fur regions are segmented, their skeleton is extracted and used to locate narrow branches which have a potential for stable grasping. Willimon *et al.* [13] segment color image of the heap into similar regions. The average height of the regions above the table is computed from stereo images, and the highest region is selected for grasping. If the grasping fails eventually, the procedure is repeated until the garment is successfully picked from the heap. Hata *et al.* [14] and Bersch *et al.* [15] choose the highest point of the heap as the grasping point, while Maitin-Shepard *et al.* [6] choose the central point. Alenyà *et al.* [16] benchmark various grasping strategies. They also describe common grasping concepts and identify possible issues.

Our approach to picking up a single garment from the pile is similar to the methods evaluating graspability of the individual garment parts [9], [10], [12]. To ensure that only one garment is grasped, an image of the pile is segmented into regions corresponding to individual garments as in [8], [13]. However, besides commonly used color information, we also consider texture while segmenting.

B. Classification, pose estimation

In a series of works, Kita *et al.* [17]–[20] deal with category recognition and pose estimation of a hanging garment. The method is based on matching the observed garment to virtual models of hanging garments. The initial method [17] utilizes a planar mass-spring model of a pullover. The model is virtually grasped for each of 20 predefined grasping points, and its resulting geometry is obtained by means of physics-simulation. Silhouette of the observed pullover is then overlaid over all simulated models to find the best matching one. The method is improved [18] by observing the garment from two different angles and checking the consistency of the individual pose estimations. Moreover, the garment category estimation is achieved by finding the best matching model from a set of models for various categories of garments. Another improvement [19] consists in reconstructing 3D point

cloud of the observed garment. The cloud is then matched to virtual 3D models simulated by professional animation software Maya. The matching of point clouds and models is however performed on their 2D projections by simply checking their overlap. The perception can be further improved by active manipulation of the hanging garment [20], including its rotation and spreading.

Li *et al.* [21], [22] also simulate grasping and hanging of 3D garment models. The hanging models are observed from many viewpoints by virtual depth sensors and described as bags of words. The acquired data are used to train SVM classifier of the garment category and pose. In the testing phase, the real hanging garment is observed from many viewpoints by a depth sensor. Each view is classified separately, and it votes for the assumed garment category and pose [21]. Time performance of the method was later significantly improved [22] by merging depth maps from various views into the single volumetric model and finding the closest simulated virtual model according to a weighted Hamming distance. Another extension [23] is unfolding of the garment in series of regraspings performed by a dual-arm robot. The best matching simulated virtual model is registered to the observed garment. The registration consists of a rigid transformation followed by a non-rigid deformation of the virtual model to the observed data. The non-rigid deformation estimation is formulated as an energy minimization problem.

Willimon *et al.* [24], [25] employed a model-based approach to estimate the pose of the hanging garment. They register a deformable mesh model of the garment to a video sequence capturing the garment being held and manipulated by a human. The registration is based on minimization of the energy enforcing smoothness of the mesh, good depth and visual correspondences between the mesh and the observed garment, etc. Willimon *et al.* [13] propose a method for estimating the category of the hanging garment which was lifted by the robotic arm. Images of the garment are matched to training images in order to find the most similar one. The similarity measure combines global features extracted from silhouettes with information about the detected edges.

Bersch *et al.* [15] deal with pose estimation and unfolding of a hanging T-shirt whose surface is covered with fiducial markers. The T-shirt is represented by a triangulated mesh. The mesh makes it possible to measure geodesic distances of the observed markers. The distances are used for pose estimation. The unfolding strategy utilizes a greedy approach of grasping the point closest to the shoulder location which is reachable.

Our approach for classification and pose estimation of the hanging garment differs significantly from the existing model based approaches [19], [22], [25], as it is driven purely by data. It uses features computed from depth maps, which are processed in the novel recognition framework of Active Random Forests. Their main advantages are efficient learning and good generalization ability, provided enough data.

C. Unfolding

Hamajima and Kakikura [7], [8] proposed a method for regrasping the lifted garment by its hemlines. Detection of the hemlines is based on the observed shadows and the shape of the hanging garment. The goal is to hold the garment at two different hemlines or two endpoints of the same hemline. Then the garment can be unfolded by its stretching.

Maitin-Shepard *et al.* [6] propose a method for unfolding and folding a heap of crumpled towels. The unfolding of a single towel is based on robust visual detection of corners. The goal is to hold the towel for its two corners sharing an edge. The towel is then untwisted by pulling it taut and rotating the grippers, pulled across the edge of the table and folded. Cusumano-Towner *et al.* [11] improved the unfolding method and they extended it for various types of garments. The garment is at first manipulated to an arbitrary recognizable configuration by regrasping its lowest point repeatedly. The sequence of regrasping operations is modeled by hidden Markov model having the garment category and the currently grasped points as hidden states. Once the garment category and not fully unfolded pose are known, the garment is laid on a table and unfolded in another series of manipulations.

Our method for unfolding the garment is similar mainly to [11]. We also model the uncertainty about the garment pose in a probabilistic framework. However, we are able to unfold the garment completely in the air, imitating the actions of a human. Our method also requires significantly less regrasping operations, and it is faster in general.

D. Flattening

After unfolding a garment, there usually appear wrinkles on the cloth that makes the subsequent template matching and folding process more difficult. Therefore a flattening step is required. Willimon *et al.* [26] proposed a method for flattening with a series of actions, where the cloth is pulled away from or towards to its centroid. Flattening is performed in two phases. In the first phase, the robot moves around the cloth, pulling at individual corners every 45 degrees, removing any minor wrinkles and folds. In the second phase, depth information is employed to find grasping points and directions for removing any existing folds. Experimental evaluation was conducted using a washcloth and different starting configurations, achieving significant flattening results [26].

Sun *et al.* [27] also address flattening of a garment on a table using depth information. However, the garment is only mildly deformed by wrinkling this time. An active stereo robot head is employed to perform accurate garment surface analysis, locate the wrinkles and estimate appropriate grasping points and directions for dual-arm flattening. In our work, no depth information is employed, since deformations of the contour are detected using garment templates. Furthermore, we use a suitable brush tool that sweeps the garment along the direction of a deformation.

E. Folding

Van den Berg *et al.* [28] introduced algorithms for folding a single garment spread on a table. The garment is folded over predefined folding lines. They showed how to grasp points on the garment contour so that the hanging part of the garment is immobilized by gravity during its manipulation.

The grasping strategy is based purely on the shape of the garment contour. Miller *et al.* [29], [30] extended that work by incorporating autonomous visual perception. They fit a predefined polygonal model to the contour of the observed spread garment to recognize its pose and category (by fitting multiple models for various types of clothes). The garment pose is checked after each fold by fitting a folded model.

Li *et al.* [31] address the folding task as well. The pose of the garment is estimated by matching a properly deformed polygonal template to the segmentation mask, formulated as an energy minimization problem. The gripper folding the garment is moved on an optimized trajectory which prevents the garment from sliding. The optimization considers empirically learned material properties and friction force between the garment and the table.

Our pose estimation method of the spread garment is similar to [29], [30] in a sense of matching polygonal models to the observed contour. However, we utilize differently defined models and employ a different matching algorithm, which is significantly more time efficient.

F. Full pipeline

Compared to the proposed work, the aforementioned methods deal mainly with individual sub-tasks that need to be solved in order to fold a pile of crumpled clothes. The exception is the work proposed by Maitin-Shepard *et al.* [6], who describe folding a heap of crumpled towels. However, we also consider T-shirts and shorts which are more complex types of clothes. Cusumano-Towner *et al.* [11] also extend the unfolding method to more types of garments, but they do not deal with their folding. Moreover, our proposed pipeline is reasonably faster than [6].

G. Active vision

We employ active vision in order to quickly detect the type, the grasp points and the pose of a garment by rotating it in an optimal manner. Two of the most representative works in this field were published by Denzler *et al.* [32], who used the mutual information criterion to optimally plan the next best camera viewpoints, and Laporte and Arbel [33], who introduced an on-line and more efficient way of measuring information gain of possible next views using the Jeffrey Divergence. Willimon *et al.* [13] use a robotic arm to pick up and classify a garment from more viewpoints to improve the classification accuracy. However, there is no intelligence behind the robot actions, which are completely random. On the other hand, our approach plans the optimal next best view of the camera taking advantage of the previously captured views and optimizing the trajectory length.

III. Garment pickup from table

The first step of the pipeline is picking up a garment from the pile of crumpled garments. We propose a generic, robust and fast technique for grasping and picking up a single item of clothing from a pile containing multiple clothes. Fig. 2a shows an example of such pile. We use a low-cost depth camera to detect folds on the surface of the garment. Such folds are the most suitable grasping points even for humans. We propose measures for assessing the graspability of the detected folds, considering the gripper geometry.

(a) Input RGB image (b) Segmented point cloud

Fig. 2: a) The original RGB image and b) the point cloud with the individual segmented regions shown in various colors.

A. Detecting candidate grasp points

Our aim is to detect candidate grasping points located along the folds of the garment. The method employs a rectified depth image $\mathbf{I}$. The rectification is based on RANSAC detection of the dominant plane corresponding to the table surface. The depth image containing the distance of 3D points to the estimated plane is computed and used as our input.

Starting from the point with the strongest response, we delete points in its vicinity which have similar scale and orientation. Since folds may be considered as ridges on the 3D surface of the garment, differential geometry techniques based on surface curvature could be used to detect them. However, we have found in practice that the input images are too noisy for robust estimation of surface normals and/or second order derivatives needed by this approach. Filtering and approximation techniques may also be computationally expensive. The proposed technique is based on the *detection of curvilinear structures* in grayscale images, originally proposed in [34]. Indeed, folds may be seen as 2D lines on the image plane with a bell-shaped profile. We use the multiscale filtering technique proposed in [34] for the detection of such *ridge points*. Briefly, this consists in filtering the depth image $\mathbf{I}$ with the following non-linear filter:

$$\mathbf{I}_\sigma(\mathbf{u}) = \min\left\{\mathrm{Pos}\left((\mathbf{E}_\mathrm{l} * \mathbf{I})(\mathbf{u})\right), \mathrm{Pos}\left((\mathbf{E}_\mathrm{r} * \mathbf{I})(\mathbf{u})\right)\right\} \quad (1)$$

We define $\mathrm{Pos}(x) = x$ for $x > 0$ and $\mathrm{Pos}(x) = 0$ for $x \leq 0$. The operator $*$ denotes *convolution*. The filters $\mathbf{E}_\mathrm{l}$, $\mathbf{E}_\mathrm{r}$ are separable and steerable 2D filters consisting of the *derivative of the Gaussian filter* (DoG) applied perpendicularly to the line direction and shifted by σ, followed by the *Gaussian filter* applied along the line direction.

In practice, for efficiency reasons, instead of DoG filtering, we first filter the images with the Gaussian kernels and then compute the *Sobel* responses for the horizontal and vertical direction. For a given scale σ, the line orientation is computed locally as the eigenvector of the *Harris operator*. To determine the scale (and thus a measure of the width of the fold), we compute $\mathbf{I}_\sigma$ over a sequence of scales, selecting the scale with

the highest response for each pixel. We refer the reader to [34] for the justification and rationale behind this approach.

Non-maxima suppression of responses across the estimated line directions is applied to obtain thin skeletal lines of detect ridges. A further pruning procedure is applied to the resulting set of points to obtain a sparse set of the candidate ridge points.

B. Computing the graspability of features

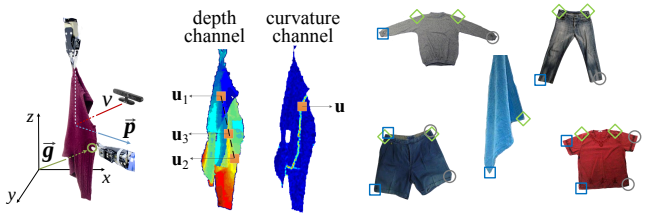
The described procedure results in a set of potential grasp points. The computed ridge position and orientation may be used to align the gripper opening with the cloth fold. An additional test is required to reject points that may result in a collision of the gripper with nearby folds, i.e. the gripper picking more than one fold at once. We also need to weight the candidates proportionally to their graspability, i.e. high and narrow folds should be preferred to shallow and/or broad ones. For this purpose, we compute the volume of the cloth inside the opening of a virtual gripper approximating the real one. The virtual gripper is placed over the candidate point, aligned with the ridge direction and left to fall until it collides with a point on the surface. We subsequently delete those candidate points where the graspability measure is too low and sort them from the most to the least graspable. Finally, an inverse kinematics test is performed to determine whether the manipulator can approach the selected point.

C. Texture segmentation

Our aim is to grasp only one garment from a pile. Therefore, a segmentation algorithm that takes account of color and texture information is necessary. To perform segmentation, we first extract *Gabor features* by convolving the RGB image with Gabor filter banks, created using multiple frequencies and orientations. The magnitude of the features is then used as a dissimilarity measure in a *graph-based segmentation* algorithm [35]. A sample of the segmentation output and the corresponding point cloud is shown in Fig. 2b. The segmentation output is subsequently combined with the results of the previous step to determine the best grasping point. In particular, we reject any regions that do not contain any candidate point. We also reject the candidate points that are too close to region boundaries. For the remaining regions and points, we sort them according to the highest (the closest one to the camera) grasp candidate point contained within their boundary. The final list of grasp candidates will contain points from the top (highest) region sorted by graspability, points from the second highest region, etc. Obviously, using texture information to segment different items has the limitations of oversegmentation (i.e. garments mixing several textures) or erroneously merging items with a similar texture. Nevertheless, the proposed approach significantly reduces the probability of grasping two items at the same item.

D. Picking

The final step is to move the manipulator so that the gripper is aligned with the candidate feature on the top of the list. The gripper comes to a certain safety distance from the table (0.5 cm), and it closes to grasp the fold. The manipulator subsequently moves to its previous position above the table. A depth image is acquired and compared to the image captured before approaching at the same position. If the average absolute difference between the images is smaller than the threshold, then grasping may have failed, and the procedure is repeated with the second candidate point in the list. Grasping was found to be very stable, and thus we did not need to detect slippage, which would require resetting the whole process. It is also not checked whether only a single garment was picked up, although the proposed approach does not guarantee this. However, we have not observed undesirable grasping of multiple garments in the performed experiments.



(a) Grasping (b) Pixel tests (c) Possible lowest points

Fig. 3: a) Grasp point $\mathbf{g}$ and pose vector $\mathbf{p}$. b) The depth and curvature channels and the random positions used in binary pixel tests. c) Blue squares and gray circles show the possible lowest points of clothes. Gray circles denote the symmetric points to the blue squared ones. Green diamonds show the desired grasping points for unfolding.

IV. Unfolding a Garment

Once a single garment is grasped and lifted up, the robot starts its unfolding. The whole unfolding procedure is performed in the air, with the help of gravity and without using a table. We detect and grasp two certain predefined points on the garment. The garment is then extended using two robotic arms, trying to imitate the actions of a human (Fig. 1b–1e).

The unfolding process consists of three main sub-problems: garment classification, grasp points detection and pose estimation, shown in Fig. 3a. We use 5 basic types of garments: long-sleeved shirts, trousers, shorts, short-sleeved T-shirts and towels. The robot first grasps the lowest point of the garment in order to reduce the number of possible configurations when it is picked up randomly. Fig. 3c shows the possible lowest points for all considered types of garments. Blue squares and gray circles denote the possible lowest points of clothes. Gray circles denote the symmetric points to the points denoted by blue squares. There are two lowest points for shorts and T-shirts, and one for shirts, trousers and towels, not counting the symmetric ones.

Each distinct possible lowest point represents a class. Furthermore, we have manually defined the grasp points for unfolding, so that the garment will unfold naturally due to gravity when grasped by these points. Fig. 3c shows the unfolding points marked by green diamonds. Thus, after classification, these points should be detected and grasped in order to unfold the garment. To facilitate the grasping, we also detect the pose of the garment. The pose of the garment is defined differently from the pose used for rigid objects. Since garments are very

deformable, the pose is difficult to be directly related to the whole shape of the garment. It is rather defined in relation to the desired grasp point, pointing to the direction of an ideal gripper approaching the desired point to grasp it properly (Fig. 3a). It is thus a property of the region around the desired point. This vector has its origin in the grasp point, it is parallel to the ground and lies on the surface plane around the desired grasp point. By estimating so defined pose, it becomes trivial to guide the gripper to grasp the desired point.

In order to jointly solve the above problems, we propose a novel unified active vision framework called *Active Random Forests* [3], which is based on classic *Random Forests* [36]. Our framework is able to perform classification and regression, actively selecting the next best viewpoint. The next subsection presents in detail the training and testing phase of Active Random Forests, formulated to solve the garment unfolding.

A. Overview

Our Active Random Forest framework is based on the classic Random Forest classifier [36]. In order to train one tree, all training samples start at the root node and keep splitting recursively into two child nodes, left and right. Splitting of samples depends on the objective function, which tries to cluster the samples, switching between a classification or regression objective. The most common metric to evaluate a split is the entropy. In order to find the best split that minimizes the objective function, there is a split function with random parameters, which is used to produce many different splits to evaluate. When samples cannot further split (due to certain criteria), the node becomes a leaf and stores a histogram of the classes of the samples arrived and/or a distribution of some continuous variables. Many trees can be trained by this way, all with different random parameters. During inference, a test sample passes down the tree, evaluating the split functions and branching left or right until it reaches a leaf node. The inference outcome is the average distribution computed from the leaf nodes of all the trees.

Based on this formulation, we introduce another property of the trees that is to be able to investigate another viewpoint of an object in an optimal way. The optimal way would be to investigate other viewpoints when the current one stops being informative according to the training samples. To this end, we introduce another type of node, called *action-selection* node. This node is created when such behavior is recognized (details are included in the next subsection). In this node, the tree decides to include another viewpoint to the process of samples splitting, again chosen in an optimal manner. In order to observe another viewpoint of a garment, the gripper holding the garment can be rotated and another image can be captured from the camera that is located at the robot base.

The following subsections describe both training and testing of our new trees in detail, including the action-selection nodes and how the trees learn to select the next best view.

B. Active Random Forests training

The training samples for Active Random Forests consist of tuples $(\mathbf{I}(v), c, \mathbf{g}(v), \mathbf{p}(v)), v \in V$, where $\mathbf{I}$ is the depth image

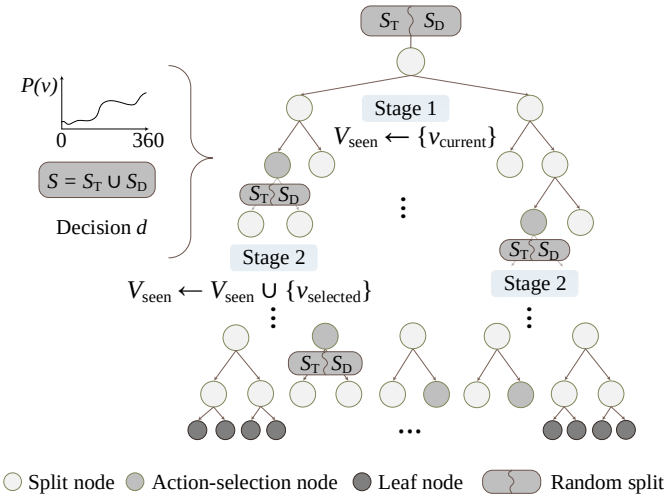


Fig. 4: Active Random Forests training procedure. The split, action-selection and leaf nodes are shown in the tree. In each action-selection node, all viewpoints are evaluated according to $P(v)$.

of the garment, c is the garment type (class), $\mathbf{g}$ is the 3D position of the desired grasp point in the current camera frame, $\mathbf{p}$ is a 3D vector of the garment pose, and V is the set of all possible viewpoints v of the garment (Fig. 3a). Viewpoints are discrete and equally distributed around the vertical axis, which coincides with the gripper holding the garment. If the desired point is not visible from viewpoint v, then $\mathbf{g}(v)$ is undefined.

Each split node of the decision tree (Fig. 4) stores a set V_{seen} of the already seen viewpoints and passes it to its children. In the beginning, only one viewpoint has been visited and therefore this set in the root node is $V_{\text{seen}} = \{v_0\}$. In each node, we evaluate a random set of split functions in the form $f(v, \mathbf{I}, \mathbf{I}_{\text{curv}}, \mathbf{u}_{\text{tripl}}) > t_{\text{split}}$ where t_{split} is a threshold. The first parameter v is a viewpoint selected randomly (uniform distribution) from the set of already visited viewpoints. The second and third parameters denote a feature channel for which the test should be performed. That is the original depth image $\mathbf{I}$ and mean curvature of the surface $\mathbf{I}_{\text{curv}}$ estimated from the depth image [2] as in Fig. 3b. The fourth parameter $\mathbf{u}_{\text{tripl}}$ is a triplet from a set of random position triples $U = \{(\mathbf{u}_1^1, \mathbf{u}_2^1, \mathbf{u}_3^1), (\mathbf{u}_1^2, \mathbf{u}_2^2, \mathbf{u}_3^2), \ldots\}$ determining positions in the image (Fig. 3b). The used binary tests are the same as in [2], which proved to be fast and efficient for our problem:

- Two pixel test $f_1 \equiv \mathbf{I}(\mathbf{u}_1) - \mathbf{I}(\mathbf{u}_2)$, where $\mathbf{I}(\mathbf{u})$ is the depth value at position $\mathbf{u}$.
- Three pixel test $f_2 \equiv (\mathbf{I}(\mathbf{u}_1) - \mathbf{I}(\mathbf{u}_3)) - (\mathbf{I}(\mathbf{u}_3) - \mathbf{I}(\mathbf{u}_2))$.
- One pixel test $f_3 \equiv |\mathbf{I}_{\text{curv}}(\mathbf{u})|$, where $\mathbf{I}_{\text{curv}}(\mathbf{u})$ is the curvature at position $\mathbf{u}$.

We want our new decision trees to perform classification, grasp point detection and pose estimation jointly. Therefore, we apply a different quality function form for each objective in the split nodes in a hierarchical coarse to fine manner [37]. That is, the classification is performed in the upper part of the trees, and when the classes have been discriminated, the lower part performs the regression of grasp points or pose vectors for each class separately. The general form of the quality function can be written as:

$$Q = \begin{cases} Q_{\text{class}}, & \text{if } \max P(c) \leq t_c \\ Q_{\text{reg}}, & \text{if } \max P(c) > t_{\text{c}} \end{cases} \quad (2)$$

Here Q_{class} is the quality function term for classification and Q_{reg} is the quality function term for regression. Specifically, Q_{class} is the *information gain* using *Shannon Entropy* and Q_{reg} is the information gain for continuous Gaussian distributions [3] weighted by the population of the nodes that a split function produces. They have the general form:

$$Q_{\text{class}} = - \sum_{\substack{child \in \\ \{\text{left,right}\}}} \frac{|S_{child}|}{|S|} \sum_{c=1}^{N_{\text{classes}}} P_{child}(c) \log_2 P_{child}(c) \quad (3)$$

$$Q_{\text{reg}} = - \sum_{\substack{child \in \\ \{\text{left,right}\}}} \frac{|S_{child}|}{|S|} \ln |\Lambda_q(S_{child})| \quad (4)$$

The set of samples in the node is denoted S, while Λ_q represents the covariance matrix of the continuous variable q being optimized. More details are provided in [38]. The form of the quality function depends on $P(c)$, which is the probability distribution of the classes of the training samples in the node, and on the predefined threshold t_{c}, typically set to 0.9. This means that if samples in a node are correctly classified, then the tree switches to regression optimization. In each node, we evaluate random split tests using (2) and keep the one maximizing Q.

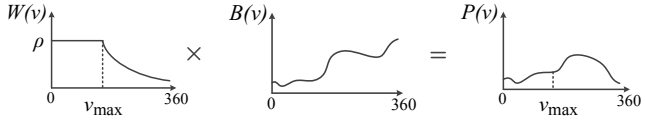
Apart from the standard objectives of classification and regression, we want to integrate the next best view selection into the decision trees, since this problem is closely related to the aforementioned objectives. In our problem, selecting the next viewpoint improves our system in two ways:

- It improves the classification and regression accuracy.
- It optimally detects a grasp point that is hidden in the current view.

Furthermore, our approach takes into account the cost of the actions, which is currently related to minimizing the execution time of the selected movements.

The split function used until now considers only the set of visited viewpoints V_{seen}. However, below certain tree depth, this set is uninformative. If the tree continues to split the samples further, it starts to overfit. In this case, a new viewpoint should be acquired to resolve the ambiguity. The problem now is to determine in advance when the current set of viewpoints is not informative. For this, we introduce a validation set, which is being split in parallel with the training set. The divergence of the posterior distributions between the two sets is measured in each split node. Specifically, the initial training set S is split into two equal-sized random subsets: S_{T} is the training set and S_{D} the validation set. Split functions are evaluated using only the training set, and when the best split function found, both sets are split using this best split function.

The comparison between the training and validation set is made in a probabilistic manner using the *Jeffrey Divergence* between the class distributions (in the case of classification) or the grasp point or pose vector distributions (in the case of regression). In the latter case, the continuous distributions are approximated with a multivariate Gaussian. More details about the resulting equations can be found in [3]. We have also tried the *Hellinger distance* as the alternative divergence measure. However, it was found that Jeffrey Divergence produced better results, so it is exclusively used in the current work.



(a) Weighted distribution (b) Visibility map (c) Final distribution

Fig. 5: Probability distributions of viewpoint used for random test selection.

If the divergence Δ between the training and validation set is above a threshold t_{Δ}, the node is considered the *action-selection node*, where an action (moving the holding gripper to see another view) should be performed to avoid overfitting. In this case, the split functions consider the whole set of possible viewpoints V of the garment which are available during training, not only V_{seen}.

To account for the execution cost of the selected actions, we assign them a cost relative to the distance from the current viewpoint (how much the gripper should be rotated), while images from the intermediate viewpoints are also captured without any additional cost. Specifically, the distance is measured as the degrees of rotation of the gripper needed to change the viewpoint. In order to weight the viewpoints according to their distance, we change the distribution from which the viewpoints v are randomly selected. The distribution of V in an action-selection node is shown in Fig. 5a. The already visited viewpoints $\{1 \dots v_{\text{max}}\}$ have a uniform distribution to be selected with probability ρ since they are not assigned any additional cost. The viewpoint v_{max} is the furthest viewpoint seen so far in the training. The next viewpoints are assigned an exponentially decreasing probability, proportional to their distance from the current view. The resulting distribution W is defined as:

$$W(v) = \begin{cases} \rho, & \text{if } v \in \{1 \dots v_{\text{max}}\} \\ \rho \exp\left(-\frac{v - v_{\text{max}}}{|V|}\right), & \text{if } v > v_{\text{max}} \end{cases} \quad (5)$$

On the other hand, the desired grasp point on the garment may be invisible in the already visited viewpoints. Therefore the next viewpoint should also make the desired point visible, apart from disambiguating the current hypotheses about the garment category or pose. The probability of the grasp point being visible can be calculated from the vectors $\mathbf{g}(v)$ in a node. The prior visibility probability $B(v)$ for each viewpoint v in node j containing samples S^j is defined as follows:

$$B(v) = \frac{\sum_{s \in S^j} b(s, v)}{\sum_{v' \in V} \sum_{s \in S^j} b(s, v')} \quad (6)$$

$$b(s, v) = \begin{cases} 1, & \text{if } \mathbf{g}_s(v) \text{ exists} \\ 0, & \text{if } \mathbf{g}_s(v) \text{ is not defined} \end{cases} \quad (7)$$

The distribution for selecting the next possible viewpoint is given by $P(v) = W(v)B(v)$. Therefore, such viewpoints, which are closer to the current one and where the grasp point

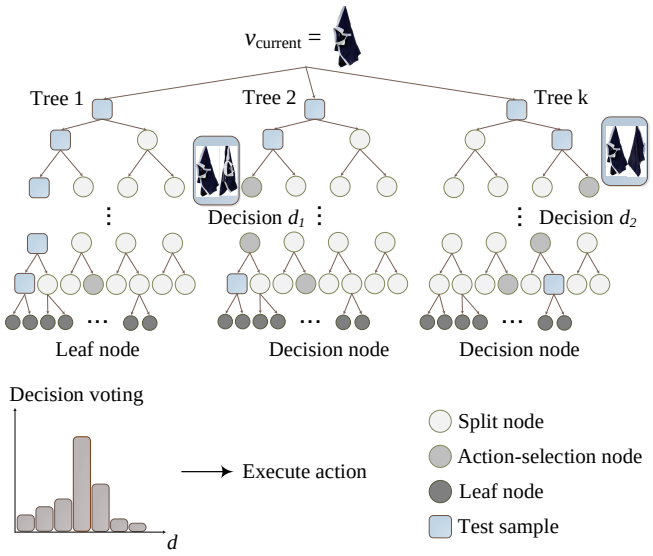


Fig. 6: Active Random Forests inference procedure. A new viewpoint is selected at each action-selection node, guiding the robot to rotate the garment. The best next viewpoint at the certain node has been found during training.

is more probable to be visible, are more likely to be selected. Fig. 5c shows an example of such distribution.

The next best viewpoint v_{best} in an action-selection node can now be found by randomly selecting viewpoints from the distribution $P(v)$. This time, the whole set of samples $S^j = S^j_{\text{T}} \cup S^j_{\text{D}}$ in the node j is used to evaluate the randomly generated tests in order to reduce the divergence between the previous training and validation set. However, in the child nodes of the action-selection nodes, the samples are again randomly split into the training and validation sets, and the process is repeated. This time, the nodes contain one more visited viewpoint, i.e. $V_{\text{seen}} = V^{\text{parent}}_{\text{seen}} \cup \{v_{\text{best}}\}$. Finally, a leaf node is created when a minimum number of samples reaches the node. In the leaf nodes, we store the class distribution $P(c)$ and the first two modes of $\mathbf{g}(v)$ and $\mathbf{p}(v)$ per class (as in [39]), weighted by the class probability, for memory efficiency.

C. Active Random Forests inference

Inference using Active Random Forests begins with the current view of the garment hanging from its lowest point. This image starts traversing the trees, evaluating the split functions selected during training. A leaf node may be reached in some trees, however, in other trees, the image ends up in an action-selection node. An another viewpoint is therefore required to continue traversing down such tree (Fig. 6). Each action-selection node from any tree votes for the next best viewpoint in a similar way, how a leaf node votes in the classical Random Forests. The most voted viewpoint is then visited and another image is captured. The trees that voted for the selected viewpoint can be further traversed using the acquired viewpoint, while the remaining trees keep their votes for the next action voting.

This process stops when N_{leafs} leaf nodes have been reached. The final class is estimated by averaging the class distributions stored in the reached leaf nodes. Grasp point detection and pose estimation are made using Hough voting of the vectors $\mathbf{g}$ and $\mathbf{p}$ stored in the leafs that we reached from all the visited viewpoints. The complete inference procedure is shown in Algorithm 1. The framework is illustrated in Fig. 6. In [3], N_{leafs} is experimentally evaluated. The outcome is that Active Random Forests require approximately twice as many trees than regular Random Forests would require, having N_{leafs} equal to the half of the total number of the trees used. This convention is also used in the current work.

Algorithm 1 Active Random Forests (ARF) inference

Input: set of pretrained trees *ARF*, current arbitrary viewpoint v_{current}
Output: garment class c, grasp point location $\mathbf{g}$, pose $\mathbf{p}$

Initialize set of seen viewpoints $V_{\text{seen}} = \{v_{\text{current}}\}$
Initialize set of reached leaf nodes $Leafs = \emptyset$
while true **do**
  Initialize *DecisionVotes* array to 0
  for all trees T in *ARF* **do**
    node $\leftarrow$ traverse tree T from node V_{seen}
    if *node* is leaf node **then**
      $Leafs \leftarrow Leafs \cup \{node\}$
      $ARF \leftarrow ARF \setminus T$
    else if *node* is action-selection node **then**
      $d \leftarrow$ viewpoint decision stored in *node*
      Increase *DecisionVotes*$[d]$
  if $|Leafs| > N_{\text{leafs}}$ **then break**
  Execute action for $d^* = \text{argmax}_d$ *DecisionVotes*$[d]$
  Update current view v_{current}
  $V_{\text{seen}} \leftarrow V_{\text{seen}} \cup v_{\text{current}}$
return average class c, Hough votes for $\mathbf{g}(v)$, $\mathbf{p}(v)$ from *Leafs*

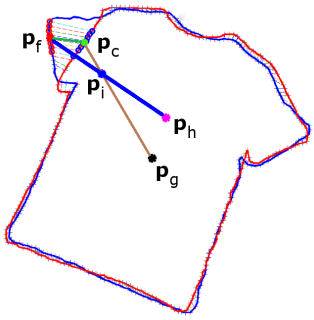
V. Spreading an unfolded garment

Once the unfolded garment is placed on the work table (Fig. 7a), it is examined in order to decide whether it is adequately spread-out for folding. This is extremely unlikely for most garments, since when unfolded they are grasped by only two points, resulting to deformations due to gravity. While experimenting with various garment types, only in case of simple geometries such as towels or shorts grasped by their waist, the robot was able to place them flat on the table.

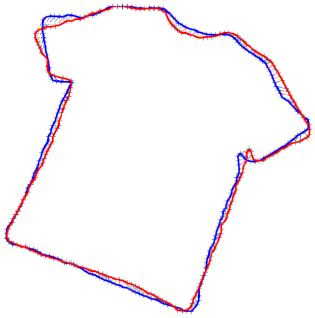
Therefore, a novel method is proposed for bringing the unfolded garment into a spread-out configuration, in case it is still deformed while placed on the work table. This method is used to bridge the gap between unfolding and folding. It is of significant importance when the complete pipeline is to be executed. Our approach is based on the measurement of the deformation between the outer contour of the examined garment and the template garment corresponding to the type recognized by the unfolding module, e.g. T-shirt, towel, shorts. In case a deformation is detected, a spreading action is executed by the robot (Fig. 7b). The spreading action consists of one arm pressing the garment towards the table in order to prevent sliding, while the other hand is sweeping with

(a) Unfolded but deformed T-shirt

(b) Fully spread T-shirt



(c) Deformed contour



(d) Contour after spreading

Fig. 7: Spreading algorithm applied on a T-shirt. a) The unfolded T-shirt has its right sleeve slightly deformed. b) Resulting configuration after spreading. c) The deformed contour (red) is matched to the garment template (blue). The spreading actions are planned based on the detected deformations. d) No deformation is detected after spreading and therefore the T-shirt can be folded.

a small brush in a suitable direction (Fig. 8). After spreading, the resulting configuration is checked again for deformations and the procedure is repeated if necessary.

In order to detect any deformations on the unfolded garment, its contour is compared to a template garment of the same type. More specifically, let $\mathbf{C}_g \in \mathbb{R}^{2\times N_c}$ denote the garment contour before spreading, whereas $\mathbf{C}_t \in \mathbb{R}^{2\times N_c}$ denotes the contour of the employed template. Both contours consist of N_c points. The contours are matched using the *Inner Distance Shape Context* (IDSC) matching algorithm [40]. The resulting correspondences are employed for estimating a similarity transformation T_s between them. The transformation T_s is then applied to $\mathbf{C}_t$, and the transformed template contour $\mathbf{C}_s = T_s\{\mathbf{C}_t\}$ is calculated.

We define the deformations as pairs of corresponding points from $\mathbf{C}_g$ and $\mathbf{C}_s$, whose distance exceeds a predefined threshold. Connected sequences of the deformed points from $\mathbf{C}_g$ are grouped together using a *sliding window*. Only the group $\mathbf{C}_{gm} \in \mathbb{R}^{2\times M_w}$, where M_w denotes the window length, having the maximum total deformation distance is selected for further processing. Apart from the distances corresponding to the magnitude of the deformations, we are also interested in the orientations of the deformations, in order to determine a suitable spreading action. Thus, all deformation vectors of $\mathbf{C}_{gm}$ are computed using the difference $\mathbf{C}_{sm} - \mathbf{C}_{gm}$, and a mean deformation vector $\mathbf{v}_m$ is also estimated.

The spreading action can be defined by three points in the image of the unfolded garment: point $\mathbf{p}_h$ denoting the *position of the holding gripper*, point $\mathbf{p}_i$ denoting the *initial position of the spreading gripper* attached a brush tool, and point $\mathbf{p}_f$ denoting the *final position of the spreading gripper* (Fig. 7c). In our approach, these points are always forming a straight line in the image plane, so that the holding arm prevents the undeformed part of the garment from moving due to the spreading motion. The estimation of these points is based on the detected position and orientation of the deformation.

Fig. 8: The brush tool attached to the gripper is moved in the direction shown by the arrow. The other arm is holding the T-shirt to prevent it from sliding.

$$\mathbf{p}_f = \mathbf{v}_m + \mathbf{p}_c \tag{8}$$

$$\mathbf{p}_i = \mathbf{p}_c + \|\mathbf{v}_m\| \frac{(\mathbf{p}_g - \mathbf{p}_c)}{\|\mathbf{p}_g - \mathbf{p}_c\|} \tag{9}$$

$$\mathbf{p}_h = \mathbf{p}_i + d_h \frac{(\mathbf{p}_i - \mathbf{p}_f)}{\|\mathbf{p}_i - \mathbf{p}_f\|} \tag{10}$$

More specifically, the central point $\mathbf{p}_c$ of $\mathbf{C}_{gm}$ is used to estimate the final position of the spreading arm using (8). The initial position of the spreading arm is estimated using (9), where $\mathbf{p}_g$ denotes the centroid of $\mathbf{C}_g$. Equation (9) implies that in order to estimate $\mathbf{p}_i$, we move from the central point of the deformation towards the centroid of the garment for a distance equal to the magnitude of $\mathbf{v}_m$. This allows starting the spreading motion from within the garment coping with deformations due to folding, whereas the spreading trajectory is shifted closer to a direction that is pointing away from the centroid. Thus, instead of using $\overline{\mathbf{p}_c\mathbf{p}_f}$, the direction of spreading is provided by the orientation of $\overline{\mathbf{p}_i\mathbf{p}_f}$. This direction is also used for estimating the position of the holding arm, which is given by (10), where d_h is a scalar, large enough to ensure that no collision takes place between holding and spreading arms. Algorithm 2 summarizes the proposed approach.

The estimated points in the image are used to determine the position of the robotic arms in world coordinates. The position of the holding arm corresponds to the end-effector, whereas the position of the spreading arm corresponds to the center of a brush tool attached to the gripper, as illustrated in Fig. 8. The orientation of the spreading arm is such that the brush tool is perpendicular to the spreading direction defined by $\overline{\mathbf{p}_i\mathbf{p}_f}$. In practice, a deviation of a few degrees can be allowed in case an inverse kinematics solution cannot be found for the estimated direction.

VI. Folding a Garment

The final step of the pipeline is folding of the garment that has been unfolded and spread on the table. Since the garment category is already known, only its pose needs to be estimated. We propose a robust method for visual detection of the garment pose from a single image. The method is not using

Algorithm 2 Contour based spreading

Input: RGB image $\mathbf{I}$ of the garment on table, template image $\mathbf{I}_\mathrm{t}$ of spread-out garment of same category as $\mathbf{I}$
Output: holding arm position $\mathbf{p}_\mathrm{h}$, spreading arm initial position $\mathbf{p}_\mathrm{i}$, spreading arm final position $\mathbf{p}_\mathrm{f}$

$d_\mathrm{max} \leftarrow 0$
$\mathbf{C}_\mathrm{g} \leftarrow$ garment contour from image $\mathbf{I}$
$\mathbf{p}_\mathrm{g} \leftarrow$ centroid of contour $\mathbf{C}_\mathrm{g}$
$\mathbf{C}_\mathrm{t} \leftarrow$ garment contour from image $\mathbf{I}_\mathrm{t}$
$T_\mathrm{s} \leftarrow$ similarity transform estimated from IDSC correspondences between contours $\mathbf{C}_\mathrm{t}$ and $\mathbf{C}_\mathrm{g}$
$\mathbf{C}_\mathrm{s} \leftarrow T_\mathrm{s}\{\mathbf{C}_\mathrm{t}\}$
$\mathbf{d}_\mathrm{Eucl} \leftarrow$ Euclidean distance of contours $\mathbf{C}_\mathrm{g}$ and $\mathbf{C}_\mathrm{s}$
if $\max(\mathbf{d}_\mathrm{Eucl}) > d_\mathrm{thresh}$ **then**
  for all points $\mathbf{p}$ in $\mathbf{C}_\mathrm{g}$ **do**
    $\mathbf{C}_\mathrm{gm} \leftarrow M_\mathrm{w}$ adjacent $\mathbf{C}_\mathrm{g}$ points with $\mathbf{p}$ at the centre
    $\mathbf{p}_\mathrm{s} \leftarrow \mathbf{C}_\mathrm{s}$ point that corresponds to $\mathbf{p}$
    $\mathbf{C}_\mathrm{sm} \leftarrow M_\mathrm{w}$ adjacent $\mathbf{C}_\mathrm{s}$ points with $\mathbf{p}_\mathrm{s}$ at the centre
    if $\|\mathbf{C}_\mathrm{sm} - \mathbf{C}_\mathrm{gm}\| > d_\mathrm{max}$ **then**
      $d_\mathrm{max} \leftarrow \|\mathbf{C}_\mathrm{sm} - \mathbf{C}_\mathrm{gm}\|$
      $\mathbf{v}_\mathrm{m} \leftarrow$ mean of $\mathbf{C}_\mathrm{sm} - \mathbf{C}_\mathrm{gm}$
      $\mathbf{p}_\mathrm{c} \leftarrow$ centroid of $\mathbf{C}_\mathrm{gm}$
  $\mathbf{p}_\mathrm{f} \leftarrow \mathbf{v}_\mathrm{m} + \mathbf{p}_\mathrm{c}$
  $\mathbf{p}_\mathrm{i} \leftarrow \mathbf{p}_\mathrm{c} + \|\mathbf{v}_\mathrm{m}\| \frac{(\mathbf{p}_\mathrm{g} - \mathbf{p}_\mathrm{c})}{\|\mathbf{p}_\mathrm{g} - \mathbf{p}_\mathrm{c}\|}$
  $\mathbf{p}_\mathrm{h} \leftarrow \mathbf{p}_\mathrm{i} + d_\mathrm{h} \frac{(\mathbf{p}_\mathrm{i} - \mathbf{p}_\mathrm{f})}{\|\mathbf{p}_\mathrm{i} - \mathbf{p}_\mathrm{f}\|}$
else
  $\mathbf{p}_\mathrm{h}, \mathbf{p}_\mathrm{i}, \mathbf{p}_\mathrm{f} \leftarrow$ NaN
return $\mathbf{p}_\mathrm{h}$, $\mathbf{p}_\mathrm{i}$, $\mathbf{p}_\mathrm{f}$

any prior information from the previous unfolding stage except the known garment category. It can thus be used separately and independently upon the pipeline, as described in [4], [5]. Once the garment pose is recognized, a single folding move is planned and executed. The vision procedure is then repeated to check the garment pose before performing the next fold.

The perception procedure can be split into several steps. It starts with a single image of the spread garment. The garment location in the image is determined by color segmentation. The garment contour is extracted from the segmentation mask. The contour is simplified by approximating it with a polygon. The simplified polygonal contour is matched to a polygonal model for the particular category of clothes. Vertices of the matched model determine locations of the important landmark points found on the garment contour, e.g. corners, shoulders, armpits or crotch. The identified landmarks are then used for the planning of the folding moves.

A. Segmentation and contour extraction

The visual sensing for estimating the garment position and configuration starts with localization of the garment on the table. The input is a single image (see Fig. 9a) taken by the camera attached to the robotic arm. We assume that color of the wooden-like table is different from the garment color and that the table color is not changing in time (except the slight changes caused by various illumination). These assumptions

(a) Input image (b) Initialization trimap (c) Segmented contour

Fig. 9: Pixels of the a) input image are used to initialize b) trimap for the GrabCut algorithm. The trimap consists of the foreground (cyan), background (yellow) and unknown (magenta) pixels. c) The garment contour (green) is extracted from the binary segmentation mask.

enable to learn a probabilistic model of the table color from training images of the table surface. The table color is modeled by the *Gaussian Mixture Model* (GMM) of RGB triples.

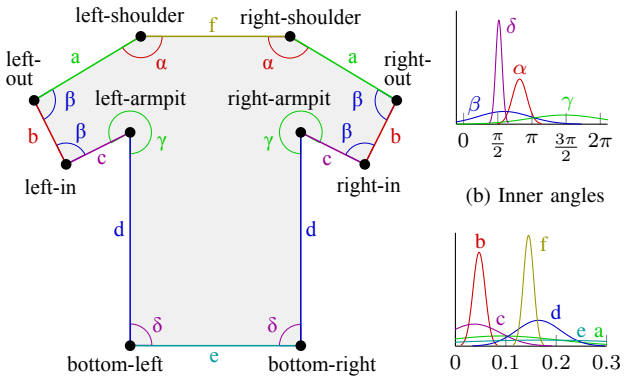
The model training starts with the initialization of GMM components using the *binary tree algorithm for palette design* [41]. The number of GMM components should be high enough to model the variability of the table color. We empirically choose three components for our wooden-like table. The prior probability, mean vector and covariance matrix for each component are learned according to the *maximum likelihood* principle [42].

While dealing with a new image depicting the garment laid on the table, we first create a preliminary hypothesis about the class of each pixel, which is based on the probability of its color in the learned GMM of table color. The highly probable pixels are labeled as background (table), the lowly probable as foreground (garment) and the mediocre ones as unknown. An example of the obtained labeling is shown in Fig. 9b. The labeling is used to initialize color models of the *GrabCut* segmentation algorithm [43] instead of requiring the user to input a sample of the foreground and background pixels. The GrabCut algorithm provides the final segmentation.

The binary segmentation mask is processed by the *Moore's algorithm* [44] for border tracking to extract the garment contour. The extracted garment contour can be seen as a polygon, which has hundreds to thousands of vertices corresponding to the individual pixels of the segmentation mask (Fig. 9c). The polygon is approximated by a new simpler polygon, which has at most dozens of vertices. The approximation method is based on running *dynamic programming algorithm for approximation of an open curve by a polyline* [45], starting from several randomly chosen vertices of the original polygon. The method does not provide the optimal approximation of the original polygon, but it is reasonably time efficient.

B. Polygonal models and their matching

The general shape of the contour for a particular clothes category in the spread state is described by a *polygonal model*, as in Fig. 10a. Each polygonal model is determined by its *vertices* and their mutual positions. The vertices were defined manually. They correspond to the important landmark points on the garment contour, e.g. corners or armpits. The admissible mutual positions of the vertices are specified by probability distributions of *inner angles* adjacent to the vertices and



(a) Polygonal model (b) Inner angles (c) Relative lengths

Fig. 10: Polygonal model for a short-sleeved T-shirt. Inner angles sharing the same distribution are denoted by the same letter $\alpha \dots \delta$. Edges sharing the same distribution of relative lengths are denoted by the same letter $a \dots f$.

probability distributions of *relative edge lengths*. Fig. 10 shows the polygonal model of a short-sleeved T-shirt. Certain angle and length distributions are shared because of the obvious left-right and top-bottom symmetries. The distributions are represented by normal density functions, which are learned from annotated training images of garments according to the *maximum likelihood* principle [42].

The polygonal model is matched to the simplified contour in order to estimate the garment pose. The matching task can be formulated as follows. The observed contour was simplified by a polygon having N vertices $\mathbf{p}_1 \dots \mathbf{p}_N$, which we will call points from now on. The polygonal model is determined by vertices $w_1 \dots w_M$ where M is specific for the particular model, e.g. it is 4 for a towel having 4 corners. It always holds $N > M$, i.e. the simplified contour comprehends more points than is the number of the model vertices.

The goal is to find mapping g of the contour points to the model vertices $g : \{\mathbf{p}_1 \dots \mathbf{p}_N\} \to \{w_1 \dots w_M, e\}$ satisfying:

- For each vertex w_m there exists a point $\mathbf{p}_i$ mapped to it.
- No two points $\mathbf{p}_i$ and $\mathbf{p}_j$ can be mapped to the same vertex w_m. However, many points can remain unmapped to any vertex. They are instead mapped to the model edge represented by the special symbol e.
- The mapping preserves the ordering of the contour points and the model vertices in the clockwise direction.
- The points are mapped to such vertices that the observed contour fits the learned model distributions of inner angles and edge lengths. The optimal mapping is found by minimizing a certain *cost function* $\mathcal{C}(g)$.

The total cost $\mathcal{C}(g)$ is given by the summation of local costs, whose definition utilizes special circular operations. Let us define the circular increment $i \oplus 1 = (i \bmod N) + 1$ and the circular decrement $i \ominus 1 = ((i-2) \bmod N) + 1$ for the point index i to remain in the set $\{1 \dots N\}$. Accordingly, we define $m \boxplus 1 = (m \bmod M) + 1$ and $m \boxminus 1 = ((m-2) \bmod M) + 1$ for the vertex index m to remain in the set $\{1 \dots M\}$.

The first of the local costs is the *vertex matching cost* $W^m_{i,j,k}$, which is defined for each triple of contour points $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k$ and each polygonal model vertex w_m:

$$W^m_{i,j,k} = -\lambda_W \log \mathcal{N}(|\angle \mathbf{p}_i \mathbf{p}_j \mathbf{p}_k|; \mu_m, \sigma^2_m) \tag{11}$$

It expresses how the size of the oriented angle $|\angle \mathbf{p}_i \mathbf{p}_j \mathbf{p}_k|$ fits the normal distribution $\mathcal{N}(\,\cdot\,; \mu_m, \sigma^2_m)$ of inner angles adjacent to the vertex w_m. The distribution mean μ_m and variance σ^2_m are learned from data. The cost is weighted by λ_W.

The *edge matching cost* $E^m_{j,k}$ is defined for each pair of points $\mathbf{p}_j, \mathbf{p}_k$ and each model vertex w_m:

$$E^m_{j,k} = -\lambda_E \log \mathcal{N}\left(\frac{\|\overline{\mathbf{p}_j \mathbf{p}_k}\|}{\sum_{i=1}^{N} \|\overline{\mathbf{p}_i \mathbf{p}_{i\oplus 1}}\|}; \nu_m, \tau^2_m \right) \tag{12}$$

It expresses how the relative length of the line segment $\overline{\mathbf{p}_j \mathbf{p}_k}$ (with respect to the overall length of the contour) fits the distribution of relative lengths of the edge $\overline{w_m w_{m\boxplus 1}}$. The distribution mean ν_m and variance τ^2_m are learned from data.

The *segment matching cost* $S_{j,k}$ is defined for each $\mathbf{p}_j, \mathbf{p}_k$:

$$\begin{aligned} S_{j,k} &= -\lambda_S \sum_{i \in I_{j,k}} \log \mathcal{N}\left(|\angle \mathbf{p}_{i\ominus 1} \mathbf{p}_i \mathbf{p}_{i\oplus 1}|; \pi, \frac{\pi^2}{16} \right) \\ I_{j,k} &= \begin{cases} \{j+1 \dots k-1\}, & \text{if } j \le k \\ \{j+1 \dots N, 1 \dots k-1\}, & \text{if } j > k \end{cases} \end{aligned} \tag{13}$$

It represents the penalty paid for points not matched to any vertex. Such points together with their neighboring segments should resemble straight lines. Thus each angle $\angle \mathbf{p}_{i\ominus 1} \mathbf{p}_i \mathbf{p}_{i\oplus 1}$ should resemble a straight angle. This is why the mean and the variance of the normal distribution are set to π and $\pi^2/16$.

The weights of the vertex, edge and segment costs were set empirically as $\lambda_W = 1$, $\lambda_E = 1/3$ and $\lambda_S = 1$ to balance their typical values. The total cost $\mathcal{C}(g)$ is given by summation of all local costs (11)–(13) with respect to the mapping g:

$$\begin{aligned} \mathcal{C}(g) = \sum \big\{ & W^m_{i,j,k} + E^m_{j,k} + S_{j,k} \mid m \in \{1 \dots M\}, \\ & g(\mathbf{p}_i) = w_{m\boxminus 1}, g(\mathbf{p}_j) = w_m, g(\mathbf{p}_k) = w_{m\boxplus 1} \big\} \end{aligned} \tag{14}$$

The number of all possible mappings of N contour points to M model vertices is combinatorial in N, M. In order to avoid their exhaustive evaluation while finding the minimum cost mapping, we propose a dynamic programming algorithm having a polynomial time complexity $O(N^5 M)$. Since $N \le 20$ points are always enough to approximate the contour accurately, the actual time performance is sufficient (see Section VII-B) despite the high degree of the polynomial.

Algorithm 3 lists the main part of the method. It finds the cost of the optimal mapping of the points $\bar{\mathbf{p}}_1 \dots \bar{\mathbf{p}}_N$ to the vertices $w_1 \dots w_M$ with respect to the condition that $g(\bar{\mathbf{p}}_1) = w_1$ and $g(\bar{\mathbf{p}}_r) = w_M$ where $r \in \{M \dots N\}$, i.e. the points mapped to the first and last vertex are given. The points $(\bar{\mathbf{p}}_1 \dots \bar{\mathbf{p}}_N) \equiv (\mathbf{p}_{n-1} \dots \mathbf{p}_N, \mathbf{p}_1 \dots \mathbf{p}_n)$ are obtained by shifting the original points by n positions. The costs $\bar{W}^m_{i,j,k}$, $\bar{E}^m_{i,j}$, $\bar{S}_{i,j}$ are shifted accordingly from $W^m_{i,j,k}$, $E^m_{i,j}$, $S_{i,j}$. Algorithm 3 is called for each shift of the contour points by n positions (out of N shifts) and for each selection of the point $\bar{\mathbf{p}}_r$ mapped to the vertex w_M (out of $N - M + 1$ selections) in order to find the cost of the globally optimal mapping with respect to (14).

Algorithm 3 Optimal mapping of contour to model

Input: point index r such that $g(\bar{\mathbf{p}}_r) = w_M$, cost $\bar{W}^m_{i,j,k}$ of matching angle $\angle\bar{\mathbf{p}}_i\bar{\mathbf{p}}_j\bar{\mathbf{p}}_k$ to vertex w_m, cost $\bar{E}^m_{j,k}$ of matching segment $\overline{\bar{\mathbf{p}}_j\bar{\mathbf{p}}_k}$ to edge $\overline{w_m w_{m+1}}$, cost $\bar{S}_{j,k}$ of approximating $\bar{\mathbf{p}}_{j+1}\dots\bar{\mathbf{p}}_{k-1}$ by segment $\overline{\bar{\mathbf{p}}_j\bar{\mathbf{p}}_k}$

Output: cost $\bar{T}^m_{j,k}$ of matching sub-contour $\bar{\mathbf{p}}_1\dots\bar{\mathbf{p}}_{k-1}$ to model vertices $w_1\dots w_m$ so that $g(\bar{\mathbf{p}}_j) = w_m$ and $g(\bar{\mathbf{p}}_k) = w_{m+1}$

for all $j \in \{2\dots r-M+2\}$ **do**
  for all $k \in \{j+1\dots r-M+3\}$ **do**

$$\bar{T}^2_{j,k} \leftarrow \left(\bar{W}^1_{r,1,j} + \bar{W}^2_{1,j,k}\right) + \left(\bar{E}^m_{r,1} + \bar{E}^1_{1,j} + \bar{E}^2_{j,k}\right) + \left(\bar{S}_{r,1} + \bar{S}_{1,j} + \bar{S}_{j,k}\right)$$

for all $m \in \{3\dots M-1\}$ **do**
  for all $j \in \{m\dots r-M+m\}$ **do**
    for all $k \in \{j+1\dots r-M+m+1\}$ **do**

$$\bar{T}^m_{j,k} \leftarrow \bar{E}^m_{j,k} + \bar{S}_{j,k} + \min_{i\in\{m-1\dots j-1\}} \left(\bar{T}^{m-1}_{i,j} + \bar{W}^m_{i,j,k}\right)$$

return $\bar{T}^M_{r,1} \leftarrow \min_{i\in\{M-1\dots r-1\}} \left(\bar{T}^{m-1}_{i,r} + \bar{W}^m_{i,r,1}\right)$

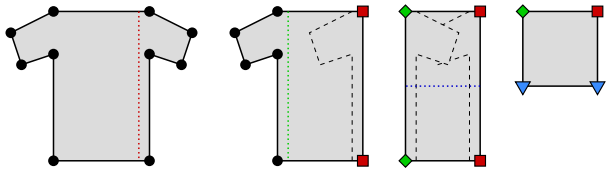


Fig. 11: Incremental creation of folded models for a short-sleeved T-shirt. The original vertices are being replaced by new vertices denoting endpoints of the individual folds (plotted with various shapes and colors).

C. Robotic folding

Once the polygonal model is matched to the observed garment contour, its pose is known and a folding move can be planned and performed by the robot. The pose of the garment after the folding move can be predicted, but we rather check it visually in order to deal with various translations and rotations of the garment caused by its manipulation. The pose estimation procedure follows the described steps. However, the simplified contour is now matched a *folded polygonal model* derived automatically from the original model of the spread garment. Fig. 11 shows the incremental creation of the folded models. The original model vertices are being replaced by new vertices denoting the endpoints of the particular fold. The distributions of inner angles and relative edge lengths are adjusted for the folded model, as described in [5].

Planning of the robotic motion for folding utilizes MoveIt package [46] contained in ROS. The trajectories are planned in three steps. The *RRT-Connect* algorithm [47] is used at first to schedule collision-free trajectories from one joint state to another one. The trajectory is then interpolated by generating evenly distributed points in Cartesian coordinates. Finally, the *inverse kinematic* is computed for each point to find the final trajectory, which is then sent to the robot controller.

The folding algorithm is based on special moves called *g-folds* [28]. The robot grasps the selected points of the garment and moves them along triangular paths. Each path goes up in its first half and down in the second half. At every moment of the move, the garment is split to the part laying on the table and the vertically hanging part. The grasping points are selected in such way that the hanging part is kept immobilized due to the gravity. Since our jaw-like gripper is not suitable for grasping flat garment from above, its lower finger slides under the garment and grasps it.

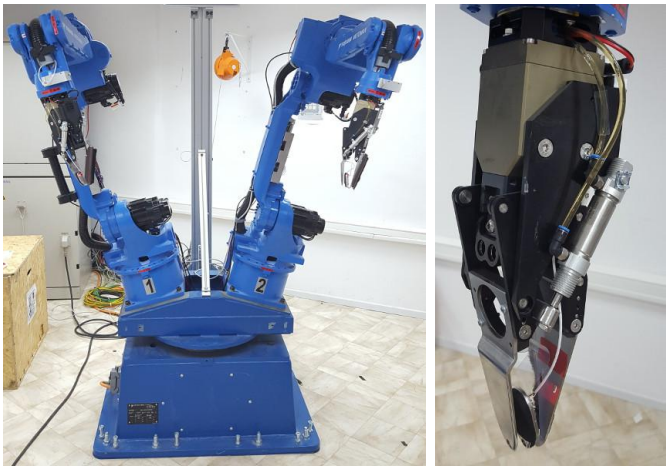


Fig. 12: The dual-arm robot used in our testbed and the detail of its arm equipped with custom jaw gripper designed for grasping of garments.

VII. Experiments

A. Testbed description

The methods described in this paper were implemented and tested on two identical dual-arm robot testbeds located in CTU Prague and CERTH Thessaloniki. The robot is composed mainly of standard industrial components. Its body consists of two Motoman MA1400 robotic arms mounted to R750 turn-table. The components are controlled by two DX100 controllers working as master and slave. The arms are attached jaw-like grippers developed by the CloPeMa consortium [48]. Fig. 12 shows a detailed view of the robot.

The robot is equipped with several sensors. There are three combined RGB and depth cameras ASUS Xtion PRO attached on the robot, two on the wrists and one on the base. They are the only sensors used for our current task. Furthermore, a head comprising of two Nikon D5100 cameras for stereo vision [49] and corresponding pan/tilt units is mounted on the robot base. The grippers are equipped with tactile sensors and photometric stereo intended mainly for material sensing. The wrists comprehend force and torque sensors.

The robot control system is built on Robot Operating System (ROS) [50] in the Hydro version. The basic functionality of moving the arms and reading positions from their joints is provided by MotoROS package, which is delivered by the manufacturer. We also utilize MoveIt package and Open Motion Planning Library (OMPL) [51] for motion planning.

B. Performance evaluation

For the evaluation of the pick-up module described in Section III, we conducted experiments on a real setup with

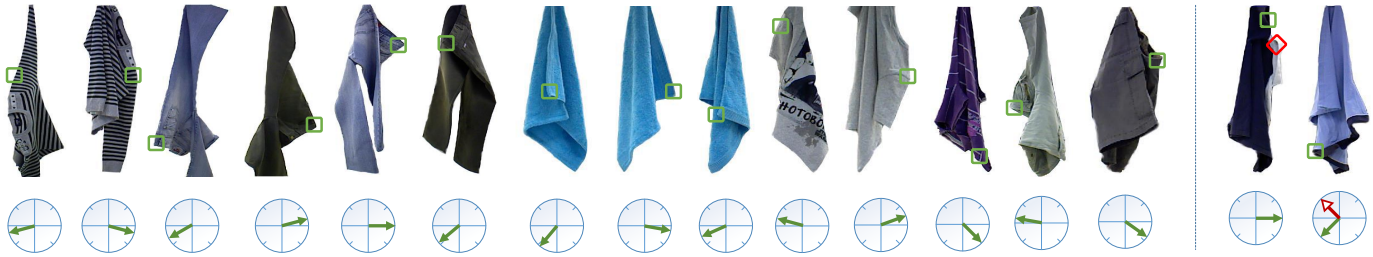


Fig. 13: Qualitative results of grasp point and pose estimation. Two failures on the right are due to wrong grasp point detection (denoted by the red diamond) and wrong pose estimation (the hollow red arrow shows the detected wrong orientation).

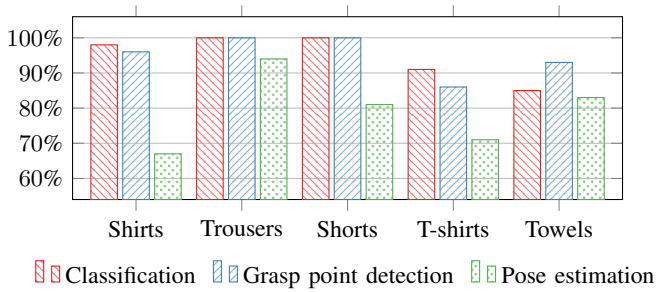


Fig. 14: Quantitative results of the garment classification, grasp point detection and pose estimation for various types of garments.

	Single		POMDP		MI		JD		Random		ARF
	SVM	RF	SVM	RF	SVM	RF	SVM	RF	SVM	RF	
Class.	72	90	91	97	84	94	93	97	88	92	**98**
Grasp	53	95	68	**97**	63	94	72	94	57	93	94
Pose	21	49	24	53	26	55	27	55	24	52	**71**

TABLE I: Comparison of Active Random Forests with three state of the art methods (POMPD, MI, JD) and two baseline methods (single and random view), all of them using two different classifiers (SVM and Random Forests). Accuracies (in percents) of all compared methods were evaluated in three tasks: classification, grasp point detection and pose estimation.

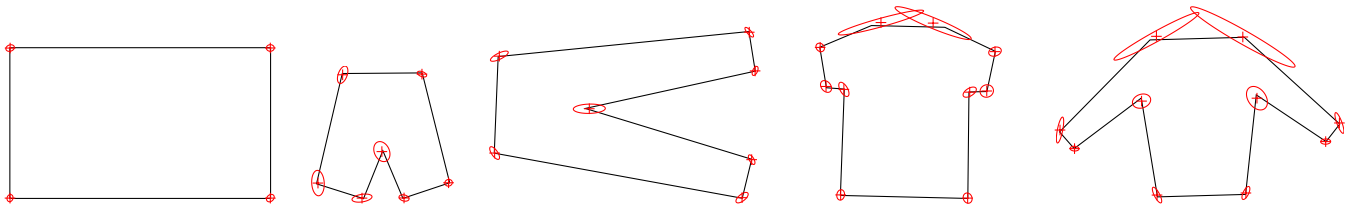
the robot repeatedly picking items from a heap. We performed 80 runs of the task. The garments in the heap were changed every 10 runs. We manually shuffled the garments on the table between the changes in order to test different configurations. The percentage of successful task completions was used as the evaluation metric, where success means picking up a single item only and lifting it above the table without significantly affecting the heap. The maximum number of 3 retries was allowed. We have achieved an overall success rate of 95 %. Half failures were caused by incorrect localization of the grasp point, whereas the remaining failures were due to slippage of the fold while the gripper was closing. The ridge detection algorithm produced biased fold locations when folds were asymmetric. Some corners were also falsely detected as ridges. Although corners can potentially be good grasping points, the noisy depth map at the vicinity of the discontinuities contributes to localization errors.

The unfolding procedure described in Section IV was tested on 30 garments, 6 per category, including long-sleeved shirts, trousers, shorts, short-sleeved T-shirts and towels. Most garments are included in the datasets of [2], [3]. Each garment was used 20 times in training, each time regrasping it from a possible lowest point. There are 72000 training images and 600 test images in total. Different garments were used for training and testing. We assume a correct grasp point detection if it is at most 10 cm close to the ground truth, whereas 18 degrees divergence is allowed for a correct pose estimation. Fig. 14 shows the accuracy of classification, grasp point detection and pose estimation for each category. We have achieved almost perfect results in classification and grasp point detection for long-sleeved shirts, trousers and shorts. Short-sleeved T-shirts and towels often look very similar in depth channel and their rate is reduced because of mutual misclassification. It can also be seen that pose estimation is the most difficult objective with less accuracy compared to the previous two. However, as we will show, ARF has significantly increased the performance compared to the state of the art in terms of the pose estimation.

We compared ARF to three baseline active vision methods: *partially observable Markov decision process* (POMDP) used in [2], *mutual information* criterion (MI) [32], and *Jeffrey divergence* metric (JD) [33]. We also compared all methods to the *random strategy*. For a fair comparison, we assumed that all actions have equal costs, which means that the prior probability distribution for selecting next viewpoints is uniform. Since the aforementioned state of the art methods require an additional classifier, we used two additional baselines for each active vision technique: Random Forests based classifier [2] and *multi-class and regression SVM* [52], [53]. To train these classifiers, we used the raw depth images of the garments as well as the HOG features [54] computed from the depth maps.

Table I shows the results for classification, grasp point detection and pose estimation. Generally, if the SVM classifier was used, the performance was the worst. Furthermore, if the single-view methods performed well, which is the case of classification and grasp point detection, the active vision methods had a positive impact on the final result, comparable to ARF. In pose estimation, however, the state of the art active vision techniques cannot improve the performance significantly, since they depend largely on single-view estimations. On the other hand, ARF has been trained to choose the most informative features and views simultaneously, better exploiting the ability of the robot to actively explore new views. Table I shows that ARF outperforms other methods by almost 20 %. Further qualitative results of ARF are shown in Fig. 13.

Four garments were employed for the evaluation of the spreading module: a towel, a pair of shorts and two T-shirts. Although, in principle, the spreading algorithm can also deal with trousers and shirts, the limited robot workspace made



(a) Towel (b) Shorts (c) Trousers (d) Short-sleeved T-shirt (e) Long-sleeved shirt

Fig. 15: Distributions of displacements between the vertices of the matched models and manually annotated landmarks. The outlines correspond to the learned models (means of normal distributions of inner angles and relative lengths were used to plot the outlines). The crosses denote mean displacement for each vertex, and the ellipses correspond to the covariance matrices of the displacements (the ellipses were plotted sized up five times).

Error	Towel	Shorts	Trousers	T-shirt	Shirt
Mean [cm]	0.3	0.8	0.6	1.0	1.1
Std. dev. [cm]	0.2	0.6	0.6	1.8	1.4

TABLE II: Displacements between the vertices found by the polygonal model matching and the manually annotated landmark points for towels, shorts, trousers, short-sleeved T-shirts and long-sleeved shirts (including sweaters).

(a) Spread T-shirt (b) Deformed sweater (c) Ill-segmented shorts

Fig. 16: Example results of the pipeline for pose estimation of a spread garment. The images show the model (plotted in cyan) matched to the extracted contour (green). The pipeline is robust enough to deal with b) partially deformed garments or c) ill-segmented images (the leather waist label was assigned wrongly to the background table segment).

their manipulation rather difficult in practice. No articles of these categories were therefore considered in the evaluation. Moreover, in the case of towels and shorts, the unfolded garment resulted in a spread-out configuration that called for no spreading action. The spreading algorithm is therefore employed only in case of T-shirts, where the sleeves are highly deformed due to gravity. In order to individually evaluate the proposed spreading algorithm described in Section V, two T-shirts were employed in a series of experiments, with the robot laying them on the table. Each T-shirt was placed on the table 15 times, whereas the other was used as the unfolded template for detecting deformations. The extracted contours consisted of $N_c = 240$ points, whereas the sliding window length was set to $M_w = 11$ points. One implicit assumption of the spreading algorithm is that a valid inverse kinematic (IK) solution can be found for the estimated spreading action. However, in practice, small deviations from the estimated spreading direction can be allowed without drastically affecting the spreading outcome. Thus, $\pm 30^\circ$ deviation was allowed, and a valid IK solution was always found for every tested configuration.

Since the spreading precedes the folding in out pipeline, the spreading may be considered successful if it results in such configuration that all landmark points are undeformed and can be detected easily by the folding algorithm. Before spreading, no configuration satisfied the above criterion, whereas after spreading 83 % configurations (25 out of 30) did. Another measure employed to assess spreading performance is the maximum Euclidean distance from the desired configuration. The desired configuration was determined manually after the spreading. The average maximum distance over all trials was 14.3 cm before spreading. It dropped to 3.7 cm after spreading. Moreover, 2.8 spreading actions were needed on average in each trial before the spreading algorithm terminated.

Section VI described the method for pose estimation of the unfolded and spread garment, which is needed for its folding. The method was tested on garments of 5 categories corresponding to the various polygonal models (see Section VI-B): towel, shorts, trousers, short-sleeved T-shirt and long-sleeved shirt. The model of a long-sleeved shirt is also used for images of sweaters whose contour shape is similar. The distributions of inner angles (Fig. 10b) and relative edges lengths (Fig. 10c) for each model were learned from 20 annotated training images. The model of table color used for segmentation (see Section VI-A) was trained from a single image of the table. The testing dataset used for the evaluation contained 53 images of towels, 32 shorts, 54 trousers, 61 short-sleeved T-shirts and 90 long-sleeved shirts and sweaters.

Table II summarizes the displacements between the manually annotated ground truth landmarks and the landmarks found by the model matching algorithm. The values were averaged over all testing images and all model vertices. Fig. 15 shows displacements for individual vertices. Individual displacement vectors collected from all testing images were normalized (based on orientation and size of the particular garment) and averaged. Their covariance matrix was computed and visualized as the ellipse sized up five times. The mean displacement is below 1.1 cm for each garment category (Table II), which is sufficient for the folding task. The largest inaccuracy reposes in the localization of shoulders where the average displacement is 2–4 cm (Fig. 15d, 15e). The reason is that the shoulders do not form such a distinctive shape on the contour as the corners or armpits do. However, the locations of the shoulders are not used for planning of the next folding move. Fig. 16 shows example matching results. The method is robust enough to deal with not fully spread garments (see shoulders of the green sweater in Fig. 16b) or with imperfect segmentation causing a significant deformation of the extracted contour (see leather waist label of the jean shorts in Fig. 16c).

The performance of the complete pipeline was evaluated on various garments (Fig. 17), including those used for testing of the spreading module. However, we have not used long-sleeved shirts and trousers, as in spreading, because of the

Fig. 17: Images of the selected garments used for testing the complete pipeline. They are depicted on the working table in a spread-out configuration.

	Shorts	T-shirts	Towels	Total
Successful / all trials count	30 / 32	21 / 32	25 / 32	**76 / 96**
Success ratio [%]	94	66	78	**79**

TABLE III: Overall results of experiments testing the complete pipeline including grasping, category recognition, unfolding, spreading and folding.

limited workspace of the robot that does not allow the folding of such long garments. We conducted 8 trials for each of 4 garments of each category (T-shirts, shorts, towels), yielding a total of 96 trials. The complete pipeline was successful in 72 trials, yielding a success rate of 79 %. This overall rate is lower than the performance of each stage because failures can occur in different stages of the pipeline and affect the complete process. More specifically, the towel was misclassified as T-shirt 7 times, grasp points were detected and grasped for shorts 2 times, and 11 unsuccessful foldings occurred for T-shirts, from which 2 occurred because the garment was not well spread on the table after unfolding. The results are summarized in Table III. The most challenging garment type is the T-shirt, which presents the poorest success rate of 66 %, despite the corrections applied by the spreading module. Fig. 1 shows all stages of the complete unfolding process. The supplementary video shows the complete folding of one garment per category.

The execution of the whole pipeline takes 8 minutes on average, with the robot operating in moderate speed for safety reasons. This can be compared to [6] where a complete pipeline for folding towels took approximately 20 minutes per towel on average. Most time is spent by the actual movement, not by perception or reasoning. Picking up a garment from a pile (Section III) takes approximately 1 second to calculate the correct grasping point, whereas the robot completes the grasping in 20 seconds. Regarding unfolding (Section IV), the image captured by the depth sensor is analyzed in 30–40 milliseconds. On average, 5 images from different viewpoints are required to classify the garment and to estimate the grasp point and pose. The whole process of finding and picking two desired grasp points per garment takes slightly over 2 minutes for the robot to execute. Regarding spreading of the garment (Section V), the deformed contour is analyzed in 10 seconds on average. Each spreading step is executed in approximately 50 seconds by the robot. The spreading process is executed at most three times. Pose estimation of the garment being folded (Section VI), which is performed prior to folding and repeated after each fold, takes 2–5 seconds, depending on the garment type. The folding action is performed in about 30 seconds by the robot. There are 2 folds needed for towels, shorts and trousers, and 3 folds for T-shirts and shirts.

VIII. Conclusion

We have described the complete pipeline for autonomous clothes folding using a dual-arm robot. To our knowledge, this is the first work addressing all necessary sub-tasks on such variety of garments. The sub-tasks comprehend: picking up a single garment from the pile of crumpled garments, recognizing the category of the hanging garment, unfolding the garment while hanging, placing the garment roughly flat on the table, spreading it, and finally folding the spread garment in a series of folding moves. The novel methods for isolating a single garment from the pile and for spreading already unfolded garment were presented in this work for the first time. They fill the gaps between our previously published method for unfolding [2], [3] and folding [4], [5], completing the full pipeline for robotic clothes folding.

The work addresses tasks related both to the machine vision and the robotic manipulation of clothes. Both of these research areas bring very challenging issues caused mainly by the fact that clothes are soft objects. It is difficult to recognize their category or estimate their pose because of their huge deformation space. The proposed techniques address these challenges, whereas some of them could also be useful in other tasks dealing with soft objects.

The performed experiments show that the proposed methods are practically applicable for robotic clothes folding. The achieved overall success rate of 79 % is very promising, considering the complexity of the pipeline. All intermediate steps have to be performed correctly in order to fold the crumpled garment. The experiments also prove that the proposed methods are general enough to deal with various types of garments, including towels, T-shirts, and shorts.

There are several possible directions for future research. The first is an improvement of the robustness. It may be achieved either by tuning the proposed methods for better performance or by designing fail-safe strategies describing how to recover from failures. An example of the already implemented fail-safe strategy is restarting of the unfolding procedure if the garment was not correctly unfolded. Our robot is equipped with multiple sensors from which only the combined camera and range sensors are used in the pipeline. The presented algorithms rely purely on visual perception. The grasping and manipulation procedures could be improved by incorporating information coming from the force, torque and tactile sensors. Another possible extension is a real-time perception, reasoning and planning performed concurrently with the manipulation. It would allow more flexible reactions to the current situation.

The existing algorithms use no or very simple representation of the observed garment. It would be interesting to model its surface globally by polygonal meshes reconstructed from the perceived input. It might also be helpful to model dynamics of the fabric. Such methods already exist in the area of computer graphics. Folding pile of clothes is one of many real world scenarios related to clothes manipulations. The skills of the robot could be extended by ironing, hanging the garment on a clothes hanger or assisting a physically challenged person with clothing.

ACKNOWLEDGMENT

The authors were supported by the European Commission under the projects FP7-ICT-288553 CloPeMa (Clothes Perception and Manipulation) and FP7-ICT- 609763 TRADR (Long-Term Human-Robot Teaming for Robot-Assisted Disaster Response), by the Technology Agency of the Czech Republic under the project TE01020197 Center for Applied Cybernetics, and by the Grant Agency of the Czech Technical University in Prague under the projects SGS15/204/OHK3/3T/13 and SGS15/203/OHK3/3T/13. Tae-Kyun Kim was supported by EPSRC grant EP/J012106/1.

REFERENCES

[1] K. Yamazaki, R. Ueda, S. Nozawa, M. Kojima, K. Okada, K. Matsumoto, M. Ishikawa, I. Shimoyama, and M. Inaba, "Home-assistant robot for an aging society," *Proc. IEEE*, vol. 100, no. 8, pp. 2429–2441, 2012.

[2] A. Doumanoglou, A. Kargakos, T.-K. Kim, and S. Malassiotis, "Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014, pp. 987–993.

[3] A. Doumanoglou, T.-K. Kim, X. Zhao, and S. Malassiotis, "Active random forests: An application to autonomous unfolding of clothes," in *Proc. European Conf. Comput. Vision (ECCV)*, 2014, pp. 644–658.

[4] J. Stria, D. Průša, and V. Hlaváč, "Polygonal models for clothing," in *Proc. Conf. on Advances in Autonomous Robotic Systems (TAROS)*, 2014, pp. 173–184.

[5] J. Stria, D. Průša, V. Hlaváč, L. Wagner, V. Petrík, P. Krsek, and V. Smutný, "Garment perception and its folding using a dual-arm robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2014, pp. 64–67.

[6] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2010, pp. 2308–2315.

[7] K. Hamajima and M. Kakikura, "Planning strategy for task untangling laundry — isolating clothes from a washed mass," *J. Robot. Mechatron.*, vol. 10, no. 3, pp. 244–251, 1998.

[8] ——, "Planning strategy for task of unfolding clothes," *Robot. Auton. Syst.*, vol. 32, no. 2, pp. 145–152, 2000.

[9] A. Ramisa, G. Alenyà, F. Moreno-Noguer, and C. Torras, "Using depth and appearance features for informed robot grasping of highly wrinkled clothes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2012, pp. 1703–1708.

[10] ——, "FINDDD: A fast 3D descriptor to characterize textiles for robot manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2013, pp. 824–830.

[11] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O'Brien, and P. Abbeel, "Bringing clothing into desired configurations with limited perception," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2011, pp. 3893–3900.

[12] G. L. Foresti and F. A. Pellegrino, "Automatic visual recognition of deformable objects for grasping and manipulation," *IEEE Trans. Syst., Man, Cybern. C*, vol. 34, no. 3, pp. 325–333, 2004.

[13] B. Willimon, S. Birchfield, and I. Walker, "Classification of clothing using interactive perception," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2011, pp. 1862–1868.

[14] S. Hata, T. Hiroyasu, J. Hayashi, H. Hojoh, and T. Hamada, "Robot system for cloth handling," in *Proc. Annu. Conf. IEEE Industrial Electronics Society (IECON)*, 2008, pp. 3449–3454.

[15] C. Bersch, B. Pitzer, and S. Kammel, "Bimanual robotic cloth manipulation for laundry folding," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2011, pp. 1413–1419.

[16] G. Alenyà, A. Ramisa, F. Moreno-Noguer, and C. Torras, "Characterization of textile grasping experiments," in *Proc. ICRA Workshop on Conditions for Replicable Experiments and Performance Comparison in Robotics Research*, 2012, pp. 1–6.

[17] Y. Kita and N. Kita, "A model-driven method of estimating the state of clothes for manipulating it," in *Proc. IEEE Workshop on Applications of Computer Vision (WACV)*, 2002, pp. 63–69.

[18] Y. Kita, F. Saito, and N. Kita, "A deformable model driven visual method for handling clothes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2004, pp. 3889–3895.

[19] Y. Kita, T. Ueshiba, E. S. Neo, and N. Kita, "Clothes state recognition using 3d observed data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2009, pp. 1220–1225.

[20] Y. Kita, E. S. Neo, T. Ueshiba, and N. Kita, "Clothes handling using visual recognition in cooperation with actions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2010, pp. 2710–2715.

[21] Y. Li, C.-F. Chen, and P. K. Allen, "Recognition of deformable object category and pose," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014, pp. 5558–5564.

[22] Y. Li, Y. Wang, M. Case, S.-F. Chang, and P. K. Allen, "Real-time pose estimation of deformable objects using a volumetric approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2014, pp. 1046–1052.

[23] Y. Li, D. Xu, Y. Yue, Y. Wang, S.-F. Chang, E. Grinspun, and P. K. Allen, "Regrasping and unfolding of garments using predictive thin shell modeling," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 1382–1388.

[24] B. Willimon, S. Hickson, I. Walker, and S. Birchfield, "An energy minimization approach to 3D non-rigid deformable surface estimation using RGBD data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2012, pp. 2711–2717.

[25] B. Willimon, I. Walker, and S. Birchfield, "3D non-rigid deformable surface estimation without feature correspondence," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2013, pp. 646–651.

[26] B. Willimon, S. Birchfield, and I. Walker, "Model for unfolding laundry using interactive perception." in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2011, pp. 4871–4876.

[27] L. Sun, G. Aragon-Camarasa, S. Rogers, and J. P. Siebert, "Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 185–192.

[28] J. van den Berg, S. Miller, K. Goldberg, and P. Abbeel, "Gravity-based robotic cloth folding," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2011, pp. 409–424.

[29] S. Miller, M. Fritz, T. Darrell, and P. Abbeel, "Parametrized shape models for clothing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2011, pp. 4861–4868.

[30] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, "A geometric approach to robotic laundry folding," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 249–267, 2012.

[31] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen, "Folding deformable objects using predictive simulation and trajectory optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2015, pp. 6000–6006.

[32] J. Denzler and C. M. Brown, "Information theoretic sensor data selection for active object recognition and state estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 145–157, 2002.

[33] C. Laporte and T. Arbel, "Efficient discriminant viewpoint selection for active bayesian recognition," *Int. J. Comput. Vision*, vol. 68, no. 3, 2006.

[34] T. Koller, G. Gerig, G. Szekely, and D. Dettwiler, "Multiscale detection of curvilinear structures in 2-D and 3-D image data," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 1995, pp. 864–869.

[35] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[36] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[37] D. Tang, T. Yu, and T.-K. Kim, "Real-time articulated hand pose estimation using semi-supervised transductive regression forests," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 2013, pp. 3224–3231.

[38] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning." Microsoft Research, Tech. Rep. MSR-TR-2011-114, Oct 2011.

[39] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, "Efficient regression of general-activity human poses from depth images," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 2011, pp. 415–422.

[40] H. Ling and D. W. Jacobs, "Shape classification using the inner-distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 286–299, 2007.

[41] M. Orchard and C. Bouman, "Color quantization of images," *IEEE Trans. Signal Process.*, vol. 39, no. 12, pp. 2677–2690, 1991.

[42] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, 2nd ed.* Wiley, 2000.

[43] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut – interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[44] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing Using MATLAB, 2nd ed.* Gatesmark, 2009.

[45] J.-C. Perez and E. Vidal, "Optimum polygonal approximation of digitized curves," *Pattern Recogn. Lett.*, vol. 15, no. 8, pp. 743–750, 1994.
[46] "MoveIt package for ROS," http://moveit.ros.org.
[47] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 995–1001.
[48] T.-H.-L. Le, M. Jilich, A. Landini, M. Zoppi, D. Zlatanov, and R. Molfino, "On the development of a specialized flexible gripper for garment handling," *Int. J. Autom. Control Eng.*, vol. 1, no. 2, pp. 255–259, 2013.
[49] A. Schmidt, L. Sun, G. Aragon-Camarasa, and J. P. Siebert, "The calibration of the pan-tilt units for the active stereo head," in *Image Process. and Commun. Challenges 7*, ser. Advances in Intelligent Systems and Computing. Springer, 2016, vol. 389, pp. 213–221.
[50] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop on Open Source Software*, 2009.
[51] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, 2012.
[52] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani, *The elements of statistical learning, 2nd ed.* Springer, 2009.
[53] G. Guo, Y. Fu, C. R. Dyer, and T. S. Huang, "Head pose estimation: Classification or regression?" in *Proc. IEEE Int. Conf. Pattern Recogn. (ICPR)*, 2008, pp. 1–4.
[54] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recogn. (CVPR)*, 2005, pp. 886–893.